**Network Programming and Automation Basics**

Student Version



Huawei Technologies Co., Ltd.

|  |
| --- |
| **Copyright © Huawei Technologies Co., Ltd. 2020. All rights reserved.**  No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Technologies Co., Ltd.  **Trademarks and Permissions**  HW_POS_RBG_Vertical-150ppi.png and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.  All other trademarks and trade names mentioned in this document are the property of their respective holders.  **Notice**  The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.  The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied. |

|  |  |
| --- | --- |
| Huawei Technologies Co., Ltd. | |
| Address: | Huawei Industrial Base  Bantian, Longgang  Shenzhen 518129  People's Republic of China |
| Website: | <https://e.huawei.com/> |

**Huawei Certification System**

Huawei Certification follows the "platform + ecosystem" development strategy, which is a new collaborative architecture of ICT infrastructure based on "Cloud-Pipe-Terminal". Huawei has set up a complete certification system consisting of three categories: ICT infrastructure certification, platform and service certification, and ICT vertical certification. It is the only certification system that covers all ICT technical fields in the industry. Huawei offers three levels of certification: Huawei Certified ICT Associate (HCIA), Huawei Certified ICT Professional (HCIP), and Huawei Certified ICT Expert (HCIE). Huawei Certification covers all ICT fields and adapts to the industry trend of ICT convergence. With its leading talent development system and certification standards, it is committed to fostering new ICT talent in the digital era, and building a sound ICT talent ecosystem.

Huawei Certified ICT Associate-Datacom (HCIA-Datacom) is designed for Huawei's frontline engineers and anyone who want to understand Huawei's datacom products and technologies. The HCIA-Datacom certification covers routing and switching principles, basic WLAN principles, network security basics, network management and O&M basics, SDN and programmability and automation basics.

The Huawei certification system introduces the industry, fosters innovation, and imparts cutting-edge datacom knowledge.



# Network Programming and Automation Basics

## Background

New protocols, technologies, and delivery and O&M modes are emerging in the network engineering field. Conventional networks face challenges from new connection requirements, such as requirements for cloud computing and artificial intelligence (AI). Enterprises are also pursuing service agility, flexibility, and elasticity. Against this backdrop, network automation becomes increasingly important.

Network programmability and automation is to simplify network configuration, management, monitoring, and operations for engineers and improve deployment and O&M efficiency. This course is to help network engineers understand Python programming and implement network automation.

## Objectives

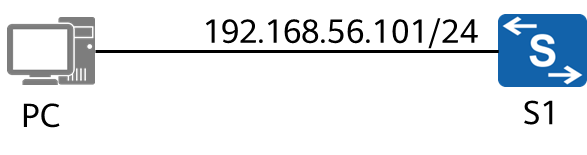
Upon completion of this task, you will be able to:

Learn the basic Python syntax

Learn how to use telnetlib

## Topology

Lab topology



A company has a switch whose management IP address is 192.168.56.101/24. You need to write an automation script to view the current configuration file of the device.

## Implementation

### Roadmap

1. Configure Telnet: Configure the Telnet password, enable Telnet, and allow Telnet access.
2. Compile a Python script: Invoke telnetlib to log in to the device and check the configuration.

### Procedure

Configure Telnet on the switch.

# Create a Telnet login password.

[Huawei]user-interface vty 0 4

[Huawei-ui-vty0-4]authentication-mode password

[Huawei-ui-vty0-4]set authentication password simple Huawei@123

[Huawei-ui-vty0-4]protocol inbound telnet

[Huawei-ui-vty0-4]user privilege level 15

Before using a Python script to log in to a device through Telnet, you need to create a Telnet password and enable the Telnet function on the device. Set the Telnet login password to **Huawei@123**.

# Enable the Telnet service to allow Telnet access.

[Huawei]telnet server enable

Info: The Telnet server has been enabled.

# Telnet to the switch from the PC using the command interface.

C:\Users\XXX>telnet 192.168.56.101

Login authentication

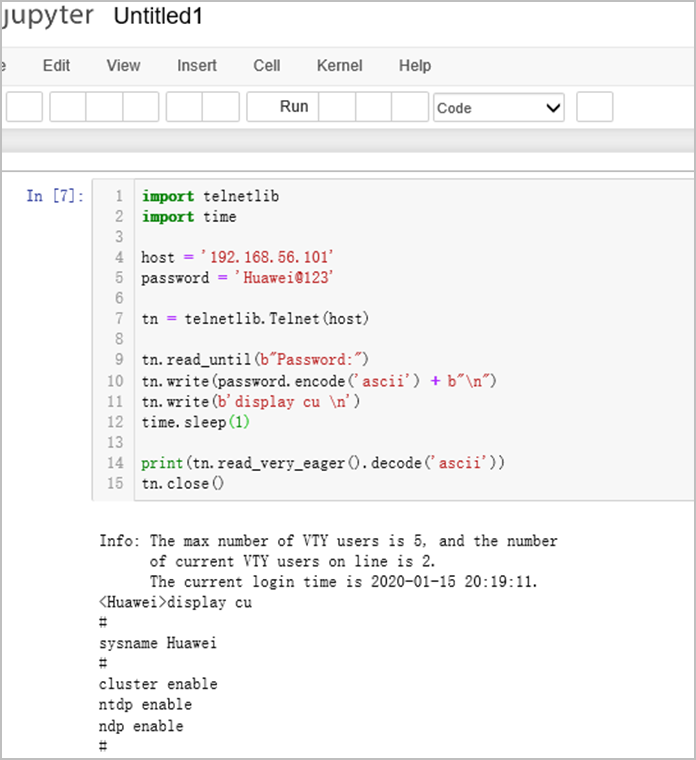
Password:

Info: The max number of VTY users is 5, and the number of current VTY users on line is 1. The current login time is 2020-01-15 21:12:57.

<Huawei>

The Telnet configuration is successful.

Execute the compiler:



The compiler used in this lab environment is Jupyter Notebook. You can also use other compilers.

Write the Python code.

import telnetlib

import time

host = '192.168.56.101'

password = 'Huawei@123'

tn = telnetlib.Telnet(host)

tn.read\_until(b"Password:")

tn.write(password.encode('ascii') + b"\n")

tn.write(b'display cu \n')

time.sleep(1)

print(tn.read\_very\_eager().decode('ascii'))

tn.close()

The Python script invokes the telnetlib module to log in to S1, runs the **display current-configuration** command, and displays the command output.

The output is as follows:

Info: The max number of VTY users is 5, and the number

of current VTY users on line is 2.

The current login time is 2020-01-15 20:19:11.

<Huawei>display cu

#

sysname Huawei

#

cluster enable

ntdp enable

ndp enable

#

drop illegal-mac alarm

#

diffserv domain default

#

drop-profile default

#

aaa

authentication-scheme default

authorization-scheme default

accounting-scheme default

domain default

domain default\_admin

local-user admin password simple admin

local-user admin service-type http

#

interface Vlanif1

ip address 192.168.56.101 255.255.255.0

---- More ----

## Code Interpretation

Import the module.

import telnetlib

import time

Import the telnetlib and time modules. The two modules are provided by Python and do not need to be installed.

This section describes the common classes and methods of the Telnetlib as the client, for example, the read\_until, read\_very\_eager(), and write() methods in the Telnet class. For more Telnet methods, see the official telnetlib document at https://docs.python.org/3/library/telnetlib.html#telnet-example.

By default, Python executes all code in sequence without intervals. When you use Telnet to send configuration commands to a switch, the switch may not respond in time or the command output may be incomplete. In this case, you can use the sleep method in the time module to manually pause the program.

Log in to the device.

Invoke multiple methods of the Telnet class in telnetlib to log in to S1.

host = '192.168.56.101'

password = 'Huawei@123'

tn = telnetlib.Telnet(host)

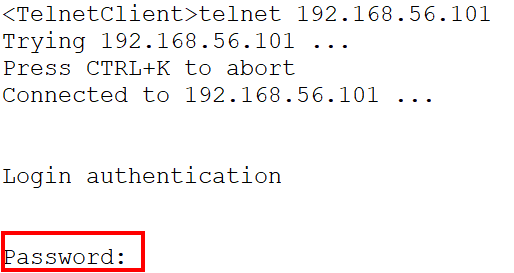
Create two variables. host and password are the login address and password of the device respectively, which are the same as those configured on the device. In this example, only the Telnet password is configured for login. Therefore, no user name is required.

telnetlib.Telnet() indicates that the Telnet() method in the telnetlib class is invoked. This method contains login parameters, including the IP address and port number. If no port information is entered, port 23 is used by default.

In this example, tn = telnetlib.Telnet(host) indicates that you log in to the device whose host is 192.168.56.101 and assign the value of telnetlib.Telnet(host) to tn.

tn.read\_until(b"Password:")

When you log in to the device at 192.168.56.101 through Telnet, the following information is displayed:



Note that the program does not know what information needs to be read. Therefore, read\_until() is used to indicate that the information in the brackets needs to be read.

In this example, tn.read\_until(b"Password:") indicates that data is read until "Password: "is displayed. The letter "b" before "Password:" indicates that the default Unicode code in Python3 is changed to bytes. This is the requirement of the function on the input data. For details, see the official document of telnetlib. If this parameter is not carried, the program reports an error.

tn.write(password.encode('ascii') + b"\n")

After Password: is displayed in the code, the program enters the password. This parameter has been defined and is used as the Telnet login password. Use write() to write the password.

In this example, tn.write (password.encode('ascii') + b"\n") consists of two parts: password.encode('ascii') and b"\n". password.encode('ascii') indicates that the encoding type of the character string Huawei@123 represented by password is ASCII. "+" indicates that the character strings before and after the symbol will be concatenated. \n is a newline character, which is equivalent to pressing Enter. Therefore, the code in this line is equivalent to entering the password Huawei@123 and pressing Enter.

Issue configuration commands.

After logging in to the device through Telnet, use the Python script to issue commands on the device.

tn.write(b'display cu \n')

write() is used to enter commands to the device. The **display cu** command is the abbreviated form of the **display current-configuration** command, which displays the current configuration of the device.

time.sleep(1)

time.sleep(1) is used to pause the program for one second to wait for the output of the switch before executing subsequent code. If the waiting time is not specified, the program directly executes the next line of code. As a result, no data can be read.

print(tn.read\_very\_eager().decode('ascii'))

print() indicates that the contents in the brackets are displayed on the console.

tn.read\_very\_eager() indicates reading as much data as possible.

. decode('ascii')) indicates that the read data is decoded to ASCII.

In this example, the code is used to display the output by S1 within one second on the console after the **display cu** command is executed.

Close the session.

tn.close()

The session is closed by invoking close(). The number of VTY connections on the device is limited. Therefore, you need to close the Telnet session after running the script.